第 39 卷第 8 期
录用定稿
计算机应用研究
Application Research of Computers
ChinaXiv合作期刊
Vol. 39 No. 8
Accepted Paper

基于数据冗余控制的移动群智感知任务分配方法 *

何杏宇 [a,b]，赵 丹 [a]，杨桂松 [a†]，金子日 [b]，覃洋恺龙 [b]，汪琦沛 [b]

(上海理工大学 a. 光电信息与计算机工程学院; b. 出版印刷与艺术设计学院, 上海 200093)

摘 要：移动群智感知系统中任务之间存在时空覆盖重叠性，这可能导致重复数据收集从而引发数据冗余问题，为此，提出了一种可同时控制任务内以及任务间数据冗余的任务分配方法。该方法首先提出基于长短期记忆(LSTM)神经网络的轨迹序列预测模型，对任务参与者进行细分时空单元的轨迹序列预测，然后根据轨迹预测结果提出最小化数据冗余的优化模型。通过最小化时空单元的数据冗余度来控制单个任务内的数据冗余问题，并通过让单个任务参与者在时空单元中的感知数据被最大化重复利用来控制多个任务之间时空覆盖重叠性带来的数据冗余。实验结果表明，所提出的任务分配方法可以有效地减少任务内及任务间的数据冗余。

关键词：移动群智感知；数据冗余；轨迹序列预测；优化模型

中图分类号：TP393 doi: 10.19734/j.issn.1001-3695.2021.12.0700

Task allocation method based on data redundant control in mobile crowd sensing

He Xingyu[a, b], Zhao Dan[a], Yang Guisong[a†], Jin Ziri[b], Qin Yangkailong[b], Wang Qipei[b]

(*a. School of Optical-Electrical & Computer Engineering*, *b. College of Communication & Art Design*, *University of Shanghai for Science & Technology*, *Shanghai* 200093, *China*)

Abstract: Due to the overlap of time and space coverage between tasks in mobile crowd sensing, repeated data collection may happen and cause data redundancy problem. In view of this, this paper proposed a task allocation method to reduce data redundancy within and between tasks. Firstly, this method designed a trajectory sequence prediction model based on the long short-term memory (LSTM) neural network, to predict trajectory sequences of task participants within subdivided spatial-temporal units. Then based on the trajectory prediction results, the method proposed an optimization model to minimize data redundancy. Specifically, the optimization model constrained the data redundancy within a single task by minimizing the data redundancy metric in each spatial-temporal unit, and limited the data redundancy between multiple tasks by maximizing the reuse of the sensing data of each task participant in a spatial-temporal unit. Experimental results verify that the proposed task allocation method can effectively reduce the data redundancy within and between tasks.

Key words: mobile crowd sensing; data redundancy; trajectory sequence prediction; optimization model

0 引言

移动群智感知(mobile crowd sensing, MCS)作为一种利用移动设备感知能力的新兴物联网感知模式，已受到学术界和产业界的广泛关注[1]。与传统的静态传感器网络相比，MCS利用嵌入在移动设备中的传感器和参与者的移动性来感知周围环境，在不花费大量成本和时间的情况下实现了更广泛的时空范围覆盖，它在环境监测[2]、城市管理[3]等场景中已有广泛应用。

感知质量和成本是移动群智感知中的重要性能指标[4]。感知质量主要由时空覆盖率衡量，成本控制一般体现在控制参与者的报酬。为了兼顾感知质量和成本，一些研究在预算限制下实现任务的全覆盖[5]或是覆盖范围的最大化[6]，另有一些研究在满足覆盖率的约束下实现感知成本的最小化[7, 8]。为了进一步控制感知成本，一些研究开始关注任务分配中的数据冗余问题。文献[9]指出，当任务达到一定的覆盖率后，再增加参与者不会过多改善感知结果，反而会增加成本。

现有的任务分配策略主要关注单个任务中的数据冗余问题。一些研究通过分析数据间的时空相关性[10, 11]，来实现高精度的数据推断，从而避免数据冗余，以降低感知成本。另一些研究考虑参与者的不确定性和不可控性，分析参与者的移动特征，来制定合理的任务分配方法。文献[12, 13]尽可能减少为同一任务服务的多个参与者之间重复的数据收集工作，来减少数据冗余。为了保证数据收集的数量并且避免每个任务收集过多的冗余数据，文献[14]定义冗余数据因子计算数据质量，文献[15]为每个任务收集的样本数量定义最大上限阈值，从而避免了任务内的数据冗余，但是忽略了任务之间的数据冗余。文献[16]为了解决参与者之间上传图片的冗余问题，提出了图相似度模型以进行细粒度的图片选择。文献[17]在参与者与任务的匹配函数中引入任务冗余因子，以惩罚多余的任务分配，从而避免了单个任务内的数据冗余并节省了预算进而有利于完成更多的任务。在压缩感知范畴中，文献[18]利用感知数据之间的隐式相关性来减少数据冗余，并选择合适的用户组以保证感知网格的时空覆盖。文献[19]研究了完全不存在数据冗余情况下的参与者选择问题，该方法假设参与者的感知区域是固定的。然而在移动群智感知场景中，参与者是移动的且参与者的移动性会对感知质量产生较大影响。

除了单个任务内部存在数据冗余外，任务之间也存在数据冗余问题。具体来说，当两个任务需要来自同一时空单元的数据时，而这个时空单元的数据收集任务被两个不同的参

收稿日期：2021-12-28；**修回日期：**2022-03-16 **基金项目：**国家自然科学基金资助项目(61802257)

作者简介：何杏宇(1984-)，女，湖南岳阳人，副教授，硕导，博士，主要研究方向为物联网、群智计算和大数据分析；赵丹(1996-)，女，山东枣庄人，硕士研究生，主要研究方向为移动群智感知；杨桂松(1982-)，男(通信作者)，河南漯河人，副教授，硕导，博士，主要研究方向为物联网与普适计算等(gsyang@usst.edu.cn)；金子日(2001-)，男，北京人，本科生，主要研究方向为移动群智感知；覃洋恺龙(2000-)，男，广西柳州人，本科生，主要研究方向为移动群智感知；汪琦沛(2001-)，男，四川峨眉山人，本科生，主要研究方向为移动群智感知.

与者重复执行将会带来任务间的数据冗余。虽然现有的移动群智感知任务分配方法有分析任务间的相关性[20,21]，其目标通常在于最小化成本。文献[20]考虑任务间的时空包含关系，旨在最小化激励成本，文献[21]定义两个任务之间的相关性指标，目的也在于降低参与者的参与成本。这些研究都忽略了任务之间的相关性对于冗余控制的影响，尤其是忽略了多个任务之间时空覆盖重叠性带来的数据冗余问题。另外，降低多个任务之间的数据冗余问题极具挑战性，因为要考虑不同任务的时空粒度，同时还要兼顾数据质量和计算激励报酬等方面的差异性。

为了实现任务分配过程中的数据冗余控制以及解决上述挑战，本文需要预测参与者未来一段时间内较长的移动轨迹序列。现有研究通常使用深度学习方法[22]、马尔可夫(Markov)模型[23]及基于参与者历史信息的概率模型或统计结果[20,24]对参与者轨迹进行预测分析。然而，实现较长轨迹序列预测时，普通循环神经网络(RNN)在解决长期依赖和上下文泛化方面存在局限性，Markov 模型随着序列长度的增加复杂度也快速增加，而基于概率模型或统计分析的方法准确度较低。相比之下，长短期记忆(LSTM)神经网络更适合有效处理长序列训练过程中的梯度消失和梯度爆炸问题，所以本文利用 LSTM 神经网络实现参与者长轨迹序列的预测。通常使用 LSTM 神经网络的轨迹预测模型将短期内的历史信息序列作为输入得到未来移动轨迹序列[25]，本文为了提高预测准确度，分析了参与者的长期轨迹信息，构建长期访问概率矩阵并作为模型的输入，使用 LSTM 神经网络并借助编码器-解码器框架实现对参与者一段时间内的连续轨迹序列预测。

综上所述，本文通过利用 LSTM 神经网络分析参与者的未来移动轨迹序列，并进一步提出一种在满足感知质量约束的同时控制任务内以及任务间数据冗余的任务分配方法。本文的主要贡献如下：

a) 为了对参与者未来移动轨迹进行更精确预测，分析了参与者的长期访问概率，定义了与参与者轨迹时空相关的访问概率矩阵，提出了基于长期短期记忆神经网络的参与者移动轨迹序列预测模型，提高轨迹预测的准确度。

b) 为了控制成本，基于参与者的移动轨迹，提出了一种最小化数据冗余度的优化模型，一方面通过细粒度时隙划分的方法最小化时空单元的数据冗余度，从而控制单个任务内的数据冗余问题；另一方面分析多个感知任务之间的时空覆盖重叠性，使参与者在时空单元中的感知数据被多个任务最大化重复利用，从而减少任务间的数据冗余，降低平台的成本。

1　系统模型和问题定义

在本系统中，一天被划分成 K 个等长的时段，所有时段的集合为 $T=\{T_1,T_2,\dots,T_K\}$，其中第 k 个时段表示为 T_k，$1\le k\le K$。将 MCS 活动区域划分成 G 个网格，所有区域的集合为 $R=\{R_1,R_2,\dots,R_G\}$，其中第 g 个分区表示为 R_g，$1\le g\le G$，网格分区易于实现且高度可扩展，可以调整网格的宽度和长度以实现不同的粒度控制。

系统中发布一系列的群智感知任务，M 个任务构成任务集 $S=\{s_1,s_2,\dots,s_i,\dots,s_M\}$，对于任意一个任务 $s_i\in S$，其都存在要求的时间范围 $T_{si}=\{T_{si}^1,\dots,T_{si}^p,\dots\}$ 及空间范围 $R_{si}=\{R_{si}^1,\dots,R_{si}^q,\dots\}$，其中 T_{si}^p 表示任务需要感知的时段，为系统划分的 T 中的任意一个指定的时段，即 $T_{si}^p\in T$，其中 $1\le p\le|T_{si}^p|$，$|T_{si}^p|$表示任务 s_i 要求感知的时段个数。R_{si}^q 表示任务需要感知的分区，为系统划分的 R 中的任意一个指定的区域，即 $R_{si}^q\in R$，其中 $1\le q\le|R_{si}|$，$|R_{si}|$为任务 s_i 要求感知的分区个数。假设系统中存在 N 个参与者 $W=\{w_1,w_2,\dots,w_j,\dots,w_N\}$，对于任意一个参与者 $w_j\in W$，都熟知系统中的任务详情，随时可以参与任务。

对于每个任务 s_i，它具有任务分配的报酬支付预算 B_{si}，并且要求参与者必须处于感知时间范围 T_{si} 及空间范围 R_{si} 内才能够为任务提供有效的感知数据。为了便于表示任务的完成情况，将任务 s_i 中的任意一个时段 T_{si}^p 与任意一个分区 R_{si}^q 形成的二元组定义为“时段-分区对”，任务 s_i 包含的时段-分区对构成的集合表示为 $TS_{si}=\{(T_{si}^p,R_{si}^q)\,|\,T_{si}^p\in T_{si},R_{si}^q\in R_{si}\}$。为了确保返回的感知结果的质量，在每个时段-分区对中需要实现收集至少 L 个感知数据报告。

为了使任务在其“时段-分区对”中收集的数据均衡分布以及降低数据冗余，现对时段进行进一步细粒度划分，将每个时段均匀划分成 L 个等长的时隙，例如将时段 T_k 划分成 L 个时隙 $t_{k(1)},\dots,t_{k(\tau)},\dots,t_{k(L)}$，其中 $t_{k(\tau)}$ 表示时段 T_k 中的第 τ 个时隙。将细分后的时隙与分区二元组命名为“时空单元”，因此一个时段-分区对包含 L 个时空单元。例如任务 s_i 的某个时段-分区对 (T_{si}^p,R_{si}^q) 经过细粒度划分后的时空单元可以表示为 $(T_{si}^p(1),R_{si}^q),(T_{si}^p(2),R_{si}^q),\dots,(T_{si}^p(\tau),R_{si}^q),\dots,(T_{si}^p(L),R_{si}^q)$，其中 $T_{si}^p(\tau)$ 表示时段 T_{si}^p 划分出的第 τ 个时隙。因此，在每个任务内，为了使收集的数据分布更加均衡，希望每个时段分区对所收集的数据都均衡的分布在其划分的每个时空单元中。另外，对于每个参与者 w_j，由于受到设备等其他因素的限制，每个参与者在系统中最多可参与 ω 个时空单元的数据收集工作。

如图 1 所示，假设任务 s_1 时间范围 $T_{s1}=\{T_2,T_3\}$，空间范围 $R_{s1}=\{R_1,R_2,R_3\}$，任务 s_2 时间范围 $T_{s2}=\{T_3,T_4\}$，空间范围 $R_{s2}=\{R_3,R_4\}$，则有时段-分区对构成的集合 $TS_{s1}=\{(T_2,R_1),(T_2,R_2),(T_2,R_3),(T_3,R_1),(T_3,R_2),(T_3,R_3)\}$，如图中绿色框和蓝色框所示，$TS_{s2}=\{(T_3,R_3),(T_3,R_4),(T_4,R_3),(T_4,R_4)\}$，如图中橙色框和蓝色框所示。以时段 T_3 为例，为了使数据在 T_3 内分布均衡，本文将 T_3 进一步划分成三个时隙 $t_{3(1)}$、$t_{3(2)}$ 和 $t_{3(3)}$，那么本文将在任务分配方法中针对每个时段-分区对包含的时空单元进行任务分配，使得收集的数据均衡分布在每个细分的时空单元内，而不是都集中于同一个时空单元，以控制每个时空单元的数据冗余。另外，任务 s_1 和 s_2 存在相同的时段-分区对 (T_3,R_3)，如图中蓝色框所示，此时时空单元 $(t_{3(1)},R_3)$、$(t_{3(2)},R_3)$、$(t_{3(3)},R_3)$ 均仅需要一份数据报告即可满足两个任务的数据收集需求。如果忽略任务 s_1 和 s_2 之间的时空重叠性，将这三个时空单元的数据收集任务独立的分配给不同的参与者，不仅会产生重复收集的数据而且会导致重复开销。本文将通过在任务分配方法中最大化每个时空单元收集数据在多个任务之间的重复利用率来降低任务之间的数据冗余。

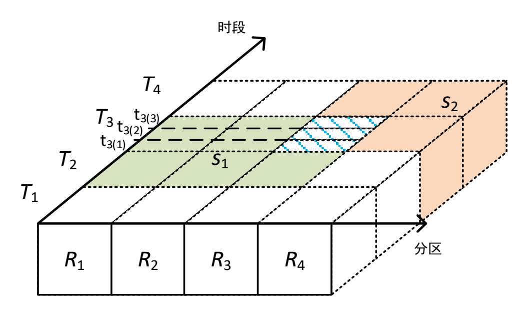


图 1　任务间时空覆盖重叠示例图

Fig. 1　Illustration of spatiotemporal coverage overlap between tasks

2　任务参与者轨迹预测模型

为了向任务分配合适的参与者，需要预测参与者在不同时空单元的移动轨迹，于是本文设计了一个基于长期短期记忆(LSTM)神经网络的参与者移动轨迹序列预测模型。相比于根据历史记录直接采用基于统计的模型来推导参与者访问时空单元的状态，本文使用 LSTM 可以实现更精确的参与者轨迹预测。预测模型如图 2 所示，包

含编码器和解码器两个部分，其中编码器用来处理数据的输入，解码器用来处理数据的输出，每个部分均含有 L 个 LSTM 单元。

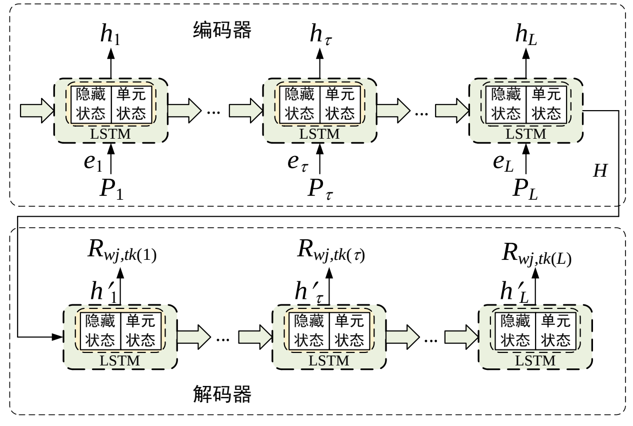


图 2　参与者轨迹预测模型

Fig. 2　Participant trajectory prediction model

由于任务参与者在历史中经常访问的地方，在未来有很大可能再次访问。为了分析影响预测轨迹序列的访问概率的特征，本文定义了与参与者轨迹时空相关的访问概率矩阵，因为历史记录中经常访问的地方，在未来有很大可能再次访问。在此基础上以形成预测模型的输入，对应图 2 中编码器部分的 P_1 到 P_L 。

访问概率矩阵表示参与者 w_j 在历史统计周期中在时段 T_k 的各个时隙中访问的各个分区的概率，用概率矩阵 P 表示。

$$P=\begin{bmatrix} p(t_{k(1)},R_1) & p(t_{k(1)},R_2) & \cdots & p(t_{k(1)},R_g) & \cdots & p(t_{k(1)},R_G) \\ p(t_{k(2)},R_1) & p(t_{k(2)},R_2) & \cdots & p(t_{k(2)},R_g) & \cdots & p(t_{k(2)},R_G) \\ \vdots & \vdots & & \vdots & & \vdots \\ p(t_{k(\tau)},R_1) & p(t_{k(\tau)},R_2) & \ldots & p(t_{k(\tau)},R_g) & \cdots & p(t_{k(\tau)},R_G) \\ \vdots & \vdots & & \vdots & & \vdots \\ p(t_{k(L)},R_1) & p(t_{k(L)},R_2) & \cdots & p(t_{k(L)},R_g) & \cdots & p(t_{k(L)},R_G) \end{bmatrix} \tag{1}$$

其中：元素 $p(t_{k(\tau)},R_g)$ 表示在历史记录中参与者 w_j 在 T_k 的时隙 $t_{k(\tau)}$ 访问分区 R_g 的概率。例如，在过去的七天历史记录中，参与者在时隙 $t_{k(1)}$ 有两次访问分区 R_1 ，有一次访问分区 R_2 ，有四次访问分区 R_3 ，则有 $p(t_{k(1)},R_1)=2/7$ ， $p(t_{k(1)},R_2)=1/7$ ， $p(t_{k(1)},R_3)=4/7$ 。

如果将矩阵 P 的第一行记为 P_1 ，第 τ 行记为 P_τ ，第 L 行记为 P_L ，则 L 个时间步的输入可以记为 $IN=\{P_1,P_2,\ldots,P_\tau,\ldots,P_L\}$ ， IN 中 L 个元素分别对应编码器中 L 个单元的输入。为了便于表示，将第 τ 个单元的输入 P_τ 记为时间步 τ 的输入，$1\le\tau\le L$ 。同时为了在不明显降低性能的情况下加快收敛过程，首先将每一个输入通过嵌入层嵌入到 θ 维的嵌入向量，时间步 τ 的输入 P_τ 的嵌入向量为 e_τ ，见公式(2)。然后将嵌入向量作为该时间步单元的输入，获得时间步 τ 的隐藏状态 h_τ ，见公式(3)。

$$e_\tau=\varphi(W_e P_\tau) \tag{2}$$

$$h_\tau=LSTM(W_{enc}[h_{\tau-1},e_\tau]) \tag{3}$$

其中：公式(2)中的 $\varphi(\cdot)$ 是由非线性激活函数 ReLU 表示的嵌入层函数，W_e 是嵌入权重。公式(3)中 e_τ 是时间步 τ 的输入，为输入信息的嵌入向量，$h_{\tau-1}$ 是上个时间步得到的隐藏状态，W_{enc} 是 LSTM 中的可学习参数，在每个输入时间步权重参数 W_{enc} 是共享的。具体来说，当前时间步 τ 的隐藏状态由上一时间步的隐藏状态和当前时间步的输入得出。

将输入时间步 1 至 L 形成的上下文向量表示为

$$H=f(h_1,h_2,\ldots,h_L) \tag{4}$$

其含义是所有输入时间步的隐含特征，由所有输入时间步的隐藏状态组成。

在预测输出阶段，在第 τ 个输出单元的隐藏状态 h'_τ 计算如下：

$$h'_\tau=LSTM(W_{dec}[h'_{\tau-1},y_{\tau-1},H]) \tag{5}$$

其中： $h'_{\tau-1}$ 表示上一输出时间步的隐藏状态， $y_{\tau-1}$ 表示上一个时间步的输出， H 是所有输入时间步形成的上下文向量，W_{dec} 表示解码器中的学习权重参数。

第 τ 个时间步的输出为

$$y_\tau=R_{wj,tk(\tau)}=softmax(h_{\tau-1}) \tag{6}$$

由此可以得出，参与者 w_j 在时段 T_k 的预测轨迹序列 $Tra_{pre}(w_j,T_k)=\{R_{wj,tk(1)},R_{wj,tk(2)},\ldots,R_{wj,tk(\tau)},\ldots,R_{wj,tk(L)}\}$ 。

3　冗余控制的任务分配方法

3.1　任务分配优化模型

本文定义参与者 w_j 关于时空单元 $(t_{k(\tau)},R_g)$ 的访问指针 $Y^{wj}_{tk(\tau),Rg}$ ，如果预测出参与者 w_j 在时段 T_k 中时隙 $t_{k(\tau)}$ 将访问分区 R_g ，则令 $Y^{wj}_{tk(\tau),Rg}=1$ ，否则为 0，即

$$Y^{wj}_{tk(\tau),Rg}=\begin{cases}1, & 如果R_{wj,tk(\tau)}=R_g\\ 0, & 否则\end{cases} \tag{7}$$

类似地，对于每个任务，用 $Y^{wj}_{T_{si}^p(\tau),R_{si}^q}=1$ 表示参与者 w_j 将在任务 s_i 的 $T_{si}^p(\tau)$ 时隙访问分区 R_{si}^q 。

若用 0-1 决策变量 $X_{ij}=1$ 表示将任务 s_i 分配给参与者 w_j ，那么任务 s_i 的时空单元 $(T_{si}^p(\tau),R_{si}^q)$ 预计收集数据样本数量如式(8)所示，任务 s_i 在时段-分区对 (T_{si}^p,R_{si}^q) 预计收集数据样本数量计算如式(9)所示，其中 $C(\cdot)$ 表示计数函数。

$$C(T_{si}^p(\tau),R_{si}^q)=\sum_{j=1}^{N}Y^{wj}_{T_{si}^p(\tau),R_{si}^q}X_{ij} \tag{8}$$

$$C(T_{si}^p,R_{si}^q)=\sum_{\tau=1}^{L}C(T_{si}^p(\tau),R_{si}^q) \tag{9}$$

对于单个任务，将任务的一个时段-分区对的数据冗余度定义为时段分区对中收集的数据数量与覆盖的时空单元数量的比值。计算如下：

$$dr(T_{si}^p,R_{si}^q)=\frac{C(T_{si}^p,R_{si}^q)}{\sum_{\tau=1}^{L}cov(T_{si}^p(\tau),R_{si}^q)} \tag{10}$$

该公式的计算前提该时段分区对已满足任务的最低感知要求。其中式(10)中的 $cov(T_{si}^p(\tau),R_{si}^q)$ 用来表示时空单元是是否被覆盖，计算如下：

$$cov(T_{si}^p(\tau),R_{si}^q)=\begin{cases}1, & 如果C(T_{si}^p(\tau),R_{si}^q)\ge 1\\ 0, & 否则\end{cases} \tag{11}$$

系统中多个任务之间可能存在时空覆盖重叠性，也就是说，多个任务之间包含相同的时段-分区对。为了降低任务间因时空覆盖重叠性带来的数据冗余，本文将让参与者 w_j 在同一时空单元感知的数据尽可能被多个任务重复利用，其重复利用度定义参与者在这一时空单元所参与的任务数量，计算如下：

$$overlap_{(tk(\tau),Rg)}(w_j)=\sum_{i=1}^{M}Y^{wj}_{tk(\tau),Rg}X_{ij} \tag{12}$$

参与者 w_j 在单个时空单元 $(t_{k(\tau)},R_g)$ 中可获得的报酬定义为单个任务支付的固定单位报酬和重叠任务经过折扣系数加权后的报酬之和，参与者在一个时空单元可获得的报酬总和由数据的重复利用度决定，计算如下：

$$pay_{(tk(\tau),Rg)}(w_j)=u+\gamma u[overlap_{(tk(\tau),Rg)}(w_j)-1] \tag{13}$$

其中： u 为单位报酬； γ 为折扣因子参数， $0\le\gamma\le 1$ 。如果参与者提供的数据被更多的任务利用，其将获得更高的报酬。

参与者 w_j 获得的报酬由重叠的任务平均支付。任务 s_i 在单个时空单元 $(T_{si}^p(\tau),R_{si}^q)$ 对参与者 w_j 支付的报酬计算如下：

$$pay_{(T_{si}^p(\tau),R_{si}^q)}(w_j,s_i)=\frac{pay_{(tk(\tau),Rg)}(w_j)}{overlap_{(tk(\tau),Rg)}(w_j)} \tag{14}$$

其中： $T_{si}^p=T_k$ 。

因此任务 s_i 关于所有时段分区的报酬支出总和为

$$pay(s_i)=\sum_{q=1}^{|R_{si}|}\sum_{p=1}^{|T_{si}|}\sum_{\tau=1}^{L}\sum_{j=1}^{N}pay_{(T_{si}^p(\tau),R_{si}^q)}(w_j,s_i) \tag{15}$$

本文的第一个优化目标为降低任务内的数据冗余，使每个时段分区对收集到数据尽可能均衡的分布在各个时空单元中。第二个优化目标为最大化每个参与者在同一时空单元感知数据

的重复利用度，即降低任务间的数据冗余。优化目标如下：

$$\begin{cases} min \quad dr(T_{si^p}, R_{si^q}) \\ max \quad overlap_{(tk(\tau),Rg)}(w_j) \end{cases} \tag{16}$$

约束为

$$pay(s_i) \le B_{si} \tag{17-1}$$

$$C(T_{si^p}, R_{si^q}) \ge L \tag{17-2}$$

$$\sum_{g=1}^{G}\sum_{k=1}^{K}\sum_{\tau=1}^{L} overlap_{(tk(\tau),Rg)}(w_j) \le \omega \tag{17-3}$$

其中：约束(17-1)表示对于每个任务 s_i，其报酬支付应不超过预算限制 B_{si}；约束(17-2)表示对于每个任务，其各个时段-分区对的收集样本数量不少于阈值 L；约束(17-3)表示每个参与者可参与的时空单元数量不得超过 ω 个。

3.2 任务分配优化模型求解

本文所要解决的问题为式(16)所示的优化问题，本文采用运行速度快且应用性较强的遗传算法来求解。由于考虑到细分的时空单元，本文问题求解空间较大，初始种群中的个体难以接近最优解，为了以较低的成本取得较好的结果，本文结合贪婪算法和遗传算法提出基于混合遗传算法的任务分配方法。在下文中，将详细地介绍本文提出的算法。

a) 分配矩阵表示。本文要解决的问题是减少数据冗余的任务分配，采用 0-1 决策变量构成的矩阵结构来表示任务分配的解，矩阵的行和列分别对应 M 个任务和 N 个参与者，M 行 N 列的分配矩阵 X 表示为 $\{X_{ij}(i=1,2,...,M; j=1,2,...,N)\}$，$X_{ij}$ 为 0 或 1。当第 i 行的第 j 个元素为 1 时，则表示任务 s_i 被分配给编号为 j 的参与者 w_j，否则不被分配。分配矩阵示例如图 3 所示。

$$\begin{array}{c} \\ s_1 \\ s_2 \\ \vdots \\ s_{M\text{-}2} \\ s_{M\text{-}1} \\ s_M \end{array}
\begin{array}{c} \begin{array}{cccccc} w_1 & w_2 & \cdots & w_j & \cdots & w_N \end{array} \\
\begin{bmatrix} 0 & 0 & \cdots & 0 & \cdots & 0 \\ 1 & 0 & \cdots & 1 & \cdots & 1 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 1 & \cdots & 1 \\ 0 & 1 & \cdots & 1 & \cdots & 1 \end{bmatrix} \end{array}$$

图 3　分配矩阵示例

Fig. 3　Allocation matrix instance

例如，假设共有 3 个任务 5 个参与者，参与者 w_1 分配给任务 s_1、s_2，w_2 分配给任务 s_1，w_3 分配给任务 s_3，w_4 分配给任务 s_2，w_5 分配给任务 s_1、s_2 和 s_3，则分配矩阵 X 可以表示为

$$X = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \end{bmatrix} \tag{18}$$

b) 初始化种群。初始群体是搜索开始时的一组染色体，其影响着算法结果的好坏，随机生成的染色体可能不能够每次都满足本文问题的约束条件。为了保证种群的多样性，本文引入贪心算子，对不满足问题约束条件的个体进行改进，如算法 1 所示。

算法 1　种群初始化

输入：任务集 S，参与者集 W，种群规模 $PopSize$。

输出：初始种群，即个体 X_1 到 $X_{PopSize}$。

a) 令 $k=1$。

b) 如果 $k > PopSize$，输出 X_k 到 $X_{PopSize}$；否则令 $W' = W$；

c) 对于 X_k 中每个元素随机以 0 或 1 赋值，令 $i=1$，$j=1$；

d) 如果 $i > M$，跳转到步骤 h)；否则对于 X_k 的第 i 行执行步骤 e)和 f)；

e) 计算每个参与者的已分配时空单元数量，如果 w_j 已分配的时空单元大于 ω，即不满足公式(17(c))，则另 $X_{ij}[i][j]=0$ 并将 w_j 从 W' 中移除；

f) 计算 s_i 的各个时段-分区对的收集数据量，如果存在时段-分区对数据量小于 L，即不满足公式(17(b))，贪婪地从 W' 中选择产生数据冗余度最小的参与者，直至不存在不满足数据量的时段分区对或 $W' = \varnothing$。

g) 令 $i=i+1$，跳转到步骤 d)；

h) 令 $k=k+1$，跳转到步骤 b)。

c) 适应度函数。本文是为了降低任务分配的数据冗余量，冗余量越小，适应性越好。因此个体 X_k 的适应度函数与数据冗余量相关，计算为

$$\frac{1}{fitness(X_k)} = \frac{DR(X_k)}{\sum_{k=1}^{PopSize} DR(X_k)} \tag{19}$$

其中：$DR(X_k)$ 表示分配方案 X_k 产生的数据冗余量。

$$DR(X_k) = DR^1(X_k) + DR^2(X_k) \tag{20}$$

其中：$DR^1(X_k)$ 和 $DR^2(X_k)$ 分别表示该分配方案在任务内及任务间产生的数据冗余量。计算公式分别如下：

$$DR^1(X_k) = \sum_{i=1}^{M}\sum_{q=1}^{|R_{s_i}|}\sum_{p=1}^{|T_{s_i}|} dr(T_{si^p}, R_{si^q}) \tag{21}$$

$$DR^2(X_k) = \sum_{g=1}^{G}\sum_{k=1}^{K}\sum_{\tau=1}^{L} \frac{max(overlap_{(tk(\tau),Rg)})}{num(t_{k(\tau)}, R_g)} \tag{22}$$

其中：$max(overlap_{(tk(\tau),Rg)})$ 表示时空单元 $(t_{k(\tau)}, R_g)$ 分配的所有参与者中最大的数据利用量。$num(t_{k(\tau)}, R_g)$ 表示该时空单元实际存在的重叠任务数量。

d) 选择。选择算子是将适应度更高的个体传递给下一代，将适应度低的个体淘汰。因为适应度低的个体也可能含有好的基因，所有本文使用轮盘赌轮盘方法来选择要保留的个体。

e) 交叉。本文以矩阵的行为单位进行部分匹配交叉操作。随机选择两个个体作为父节点并设置交叉点，然后交换两个父节点所设定交叉点所在的行生成两个新个体。如果新个体超过参与者的执行能力，则重新设置交叉点，直至满足公式(17-3)。

f) 变异。为了避免算法结果陷入局部最优并加快算法收敛速度，本文随机地在矩阵中选择变异元素，将元素“1”变为“0”，元素“0”变为“1”，同时要验证变异后的个体是否满足任务的最低需求，即式(17-2)。

g) 终止。当迭代次数达到最大进化代数时终止运行。

算法 2 为提出的任务分配方法的执行过程。

算法 2　任务分配算法

输入：任务集 S，参与者集 W。

输出：分配矩阵 X_{best}。

a) 根据算法 1 初始化种群，即产生个体 X_1 到 $X_{PopSize}$。

b) 初始化迭代次数，令 $Iter=0$。

c) 根据公式(19)计算种群中个体适应度；

d) 更新当前最佳个体 X_{best}；

e) 通过轮盘赌选择适应度高的个体；

f) 对选择的个体进行按行交叉操作；

g) 选择个体以预定义概率进行突变，将元素“1”变为“0”，元素“0”变为“1”；

h) 令 $Iter = Iter + 1$，达到迭代阈值则停止迭代；否则跳转到步骤 c)；

i) 输出最佳个体 X_{best}。

4 实验分析

针对 MCS 中的任务分配问题，本文综合考虑了任务的时空属性及感知质量要求、参与者的未来可能轨迹序列等因素，以最小化任务内的数据冗余度及最大化任务间的感知数据利用率为目标，基于遗传算法设计了一种减少感知成本的任务分配算法。为了评估所提出方法的性能，在 Python 中首先对本文的预测方法准确度进行了实验对比，然后对任务分配算法的效率进行了实验验证。实验的主要参数设置在表 1 中给出。

4.1 对比算法

为了评估本文算法，将以下两种算法(细粒度的多任务分配算法(MTPS)[15]、启发式参与者招募机制(CAPR)[21])作为基线，针对在不同任务数量上的情况进行性能比较。

细粒度的多任务分配算法(MTPS)-该算法采用集中式方法在总预算限制下，根据效用函数使用迭代贪婪的过程优化分配，以实现感知质量最大化。MTPS 中设计了合理的激励函数并对任务进行了细粒度的周期划分，是较典型的控制任务内数据冗余的方法。

启发式参与者招募机制(CAPR)-该方法考虑任务之间及参与者之间的相关性(包括正相关和负相关)进行参与者招募，提出三阶段的启发式参与者招募机制，以降低参与者成本并实现平台效用最大化。虽然 CAPR 没有直接实现任务间数据冗余控制，但该机制定义任务间的相关函数并在分配过程中考虑任务之间的相关性，可以在一定程度上可以降低任务间的数据冗余。

关于 CAPR 中定义的任意两个任务 s_i 和 $s_{i'}$ 间的相关函数 $S(s_i,s_{i'})$ 在本文中用时空覆盖重叠率衡量，具体计算见公式(23)。另外，由于本文忽略参与者的信誉因素，因此在 CAPR 中将参与者信誉值设置为常数 1。

$$S(s_i,s_{i'})=\begin{cases}0, & TS_{s_i}\cap TS_{s_{i'}}=\varnothing\\ 1, & TS_{s_i}\subseteq TS_{s_{i'}}\text{或}TS_{s_{i'}}\subseteq TS_{s_i}\\ |TS_{s_i}\cap TS_{s_{i'}}|/|TS_{s_i}\cup TS_{s_{i'}}|, & \text{其他}\end{cases}\quad(23)$$

其含义当任务 s_i 和 $s_{i'}$ 不存在时空重叠时，两者的相关系数为 0；当一个任务含于另一个任务时，两者的相关系数为 1；否则，两者的相关系数为任务的时段-分区对的交集数与并集数的比值，此时 $S(s_i,s_{i'})\in(0,1)$ 。

表 1　参数设置

Tab.1　Parameter settings

参数	取值范围
时段 T 划分个数	12
任务数 M	[20,100]
参与者数 N	15
参与者参与时空单元数量限制 ω	30
任务的预算 B_{s_i}	[100,220]
时隙划分长度 L	[2,12]

4.2 评估指标

为了验证所提出方法的有效性，本文首先对预测方法的准确性进行了实验分析。然后根据任务执行率、数据冗余率和感知成本三项指标对任务分配算法进行评估，分析任务数量的变化对三项指标的影响。

a) 任务执行率

本文将任务执行率定义为分配成功的任务数量(即满足最低感知质量的任务)与系统总任务数量的比值，任务执行率的范围为[0, 1]。

$$\text{任务执行率}=\frac{\text{分配成功的任务数量}}{\text{总任务数量}}$$

b) 数据冗余率

本文将数据冗余率定义为整体与有效数据所占比例的差，其中有效数据指所收集数据覆盖的时空单元数量，因此数据冗余率的取值范围在[0, 1]之间。

$$\text{数据冗余率}=1-\frac{\text{有效数据样本}}{\text{收集的总数据样本}}=\frac{\text{收集的总数据样本}-\text{有效数据样本}}{\text{收集的总数据样本}}$$

c) 平均感知成本

本文将平均感知成本定义为平台的报酬支付总和与已分配的任务数量的比值，表示平均每个任务的成本。

$$\text{平均感知成本}=\frac{\text{平台总支付}}{\text{已分配任务数量}}$$

4.3 实验结果分析

4.3.1 预测方法评估

为评估本文提出的参与者轨迹序列预测方法，对本文预测方法与基于统计分析的预测方法[20]、基于 Markov 模型的方法[23]和考虑短期历史信息的 LSTM 方法[25]进行了对比实验。其中文献[20]根据历史轨迹记录采用统计模型推导参与者在特定时间通过特定分区的概率，文献[23]利用半马尔可夫过程计算参与者的访问概率，文献[25]将预测时间点之前的一段历史轨迹序列作为预测模型的输入。在实验中，本文将一天划分成 T =12 个时段，对时段的不同划分个数，即 L =2、4、6、12 分别进行了准确度分析。

图 4 显示时段划分个数 L 不同取值下的预测方法准确度。从图中可以看出，本文预测方法的准确度高于其他 3 种方法。可以看出 Markov 方法和统计分析方法准确度较低，表明 LSTM 神经网络自学习功能可以有效提高预测的准确性。考虑短期信息的 LSTM 方法准确度略低于本文预测方法，其原因是本文考虑了参与者的长期移动轨迹的概率信息，更具有普遍性。另外，本文的预测方法在 L 取值为 4 时，准确度最高，约为 86.4%，随着 L 的增大，四种方法的准确度均在降低，其原因是 L 增大会使预测时间间隔变小，参与者的轨迹多变，规律不明显，因此准确性降低。

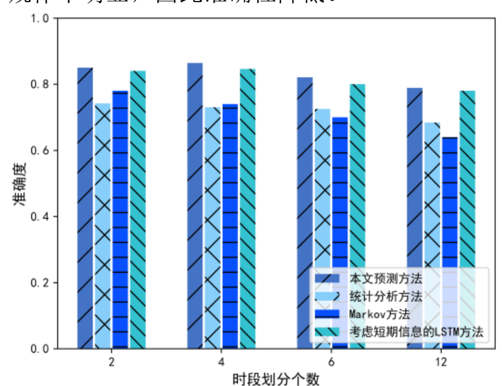


图 4　L 对预测方法准确度的影响

Fig. 4　The influence of L on the accuracy of prediction method

4.3.2 任务分配方法评估

为说明任务数量变化对实验结果的影响，本文固定参与者数量为 15，参与者可参与的时空单元数量最大限制为 30。首先比较了本文所提算法的收敛情况，然后通过改变任务的数量对本文算法、CAPR 机制和 WTPS 算法进行了在任务执行率、数据冗余率和平均感知成本三个方面的指标对比，并且每项指标均重复进行了 10 次实验取平均值作为实验结果。

表 2 展示了算法的迭代次数。为直观表示本文算法收敛性能，将本文算法与未使用算法 1(种群初始化算法)初始化的算法作关于算法收敛时的迭代次数比较。可以看出，随着任务数量的增加，本文算法和未初始化算法的迭代次数均在增加，未初始化算法的迭代次数大于本文算法。可以表明，本文提出的算法 1 有利于加快任务分配求解速度。

表 2　迭代次数

Tab. 2　Number of iterations

任务数量	20	40	60	80	100
本文算法	50	200	280	400	540
未初始化算法	70	250	350	570	820

图 5 显示任务数量变化对任务执行率的影响。可以看出，本文算法具有更高的任务执行率，WTPS 算法效果较差。其原因是，WTPS 虽然考虑按照周期分配的方式降低任务内的

数据冗余，却忽略了任务之间的时空覆盖重叠性，在任务之间存在过多的重复分配，产生较高的支付成本，不利于选择更多的参与者满足任务最低阈值要求。随着任务数量的增加，三种方法实现的任务执行率整体趋势均在下降。因为参与者受到最多可参与时空单元数量的限制，随着任务数量的增加，任务分配逐渐达到参与者的可参与时空单元数量阈值，没有足够的参与者进行任务分配。其中，本文算法与 CAPR 机制效果差距较小，因为任务内的数据冗余对任务分配成功数量影响小于任务间的影响，CAPR 即使忽略了任务内的数据冗余，但由于其可以根据任务间的相关性节约成本，因此可以有较高的任务执行率。另外，当任务数量低于 50 时，本文算法和 CAPR 的任务执行率均在 0.84 左右，当任务数量高于 50 时，本文算法的效果优于 CAPR 机制。其原因是，当任务数量较少时，任务之间的时空覆盖重叠度较低，本文算法的最大化时空覆盖重叠区域中参与者数据的重复利用度效果不明显，与 CAPR 实现结果接近；而当任务数量较多时，任务之间的时空覆盖重叠度增加，本文算法中的最大化参与者数据的重复利用度可以尽最大可能的提高时空覆盖重叠区域内任务分配的数量，因此任务执行率优于 CAPR。另外，CAPR 在任务开始时间作出的预分配可能不适合后续的分配，因为后续出现的任务负相关或者参与者不可用情况可能导致任务分配失败。

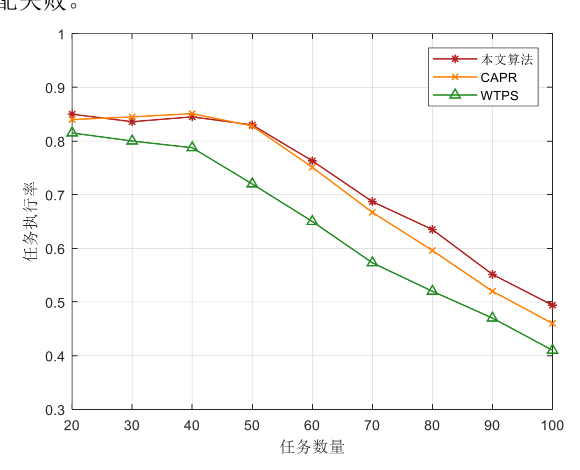


图 5　任务数量变化对任务执行率的影响

Fig. 5　The influence of changes in the number of tasks on task completion rate

图 6 显示任务数量变化对数据冗余率的影响。从图中可以看出，本文算法实现的数据冗余率低于 CAPR 机制和 WTPS 算法。其原因是 WTPS 仅是降低任务内的数据冗余，CAPR 机制考虑任务的时空相关性可以在一定程度上减少任务间的数据冗余，而本文算法一方面通过细粒度时隙划分的方法减少任务内的数据冗余问题，另一方面使参与者在时空单元中的感知数据最大化重复利用，同时实现任务内和任务间的数据冗余控制。当任务数量增多时，本文算法和 CAPR 的数据冗余率缓慢增加，而 WTPS 数据冗余率增加较快，本文算法的数据冗余率增加至 22%左右，WTPS 的数据冗余率已达到 30%，差距逐渐增大。其原因是，任务数量增加可能产生更多具有时空覆盖重叠的任务，WTPS 忽略了任务之间的时空覆盖重叠性，进行了过多的重复任务分配，产生了较多任务之间的冗余数据，且随着系统任务数量的增多，重复分配的任务增加。本文算法的数据冗余率缓慢增加，因为任务之间的时空覆盖重叠度增加，时空覆盖重叠区域中参与者的数据利用率增加，所以产生的数据冗余率变化趋势缓慢。从图中可以看出，相比较于本文算法，CAPR 的数据冗余率增加更加缓慢，其原因可能是在任务数量较少时 CAPR 在任务内部已经产生较多的冗余数据；而当任务数量增加时，由于时空覆盖重叠的任务较多，CAPR 可以在一定程度上降低任务间的数据冗余，而产生较少的冗余数据。因此 CAPR 的数据冗余变化不明显。

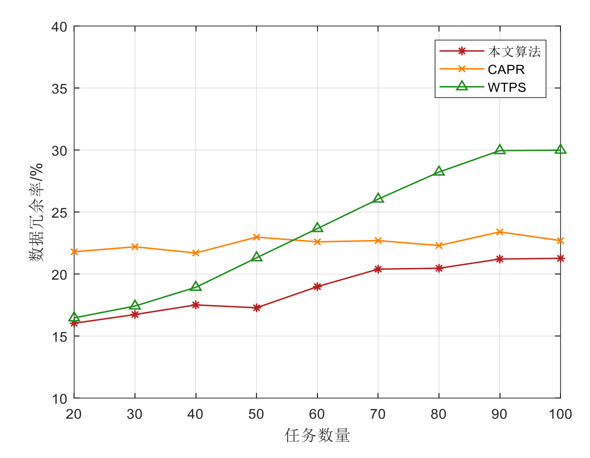


图 6　任务数量变化对数据冗余率的影响

Fig. 6　The influence of changes in the number of tasks on data redundancy rate

图 7 显示任务数量变化对平均感知成本的影响。在任务数量低于 40 时，三种方法的平均感知成本相近，但随着任务数量的增加，本文算法和 CAPR 的感知成本逐渐降低。其原因是，随着任务数量的增加，任务之间的时空覆盖重叠性增加，本文算法可以提高时空覆盖重叠区域参与者数据的重复利用度，有效降低任务的报酬支付，因此随着任务数量的增加，本文算法的平均感知成本逐渐降低。同理，CAPR 为任务选择参与者的过程中考虑任务相关性，在一定程度上可以降低感知成本，并且当任务数量较少时，CAPR 有较高的任务执行率，因此平均感知成本相较于本文算法更低。另外，可以看出 WTPS 算法的平均感知成本变化比较平稳，WTPS 没有考虑任务间的数据冗余，说明随着任务数量增加由于受到预算的限制，WTPS 并没有完成更多的任务分配。

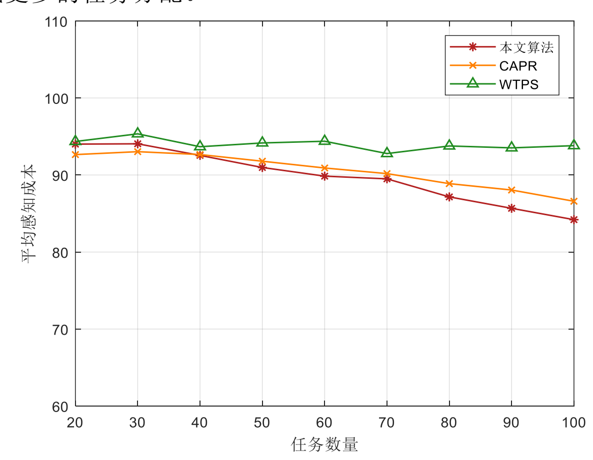


图 7　任务数量变化对平均感知成本的影响

Fig. 7　The influence of changes in the number of tasks on average sensing cost

5　结束语

移动群智感知系统中多个任务之间存在时空覆盖重叠性可能导致重复数据收集从而引发数据冗余问题，本文提出了一种可降低任务内及任务间数据冗余的任务分配方法。本文首先设计了参与者移动轨迹序列的预测方法，然后在预测结果的基础上考虑到任务之间的时间及空间范围的重叠性，提出了基于遗传算法的任务分配方法。仿真结果表明提出的方法在降低任务内及任务间的数据冗余方面有良好的效果。在

未来的工作中，应考虑更多可能影响参与者行为的因素以及参与者自身的时间可用性等因素对任务分配的影响，并探索新的优化方法和理论基础。

参考文献：

[1] Hettiachchi D, Kostakos V, Goncalves J. A survey on task assignment in crowdsourcing [J]. ACM Computing Surveys, 2022, 55 (3): 1–35.

[2] Seid S, Zennaro M, Libse M, *et al.* Mobile crowdsensing based road surface monitoring using smartphone vibration sensor and lorawan [C]// Proc of the 1st Workshop on Experiences with the Design and Implementation of Frugal Smart Objects. New York: ACM Press, 2020: 36-41.

[3] Bock F, Martino S D, Origlia A. Smart parking: using a crowd of taxis to sense on-street parking space availability [J]. IEEE Trans on Intelligent Transportation Systems, 2020, 21 (2): 496–508.

[4] 方文凤，周朝荣，孙三山. 移动群智感知中任务分配的研究 [J]. 计算机应用研究，2018，35 (11): 3206-3212. (Fang Wenfeng, Zhou Zhaorong, Sun Sanshan. Research on task assignment for mobile crowd sensing [J]. Application Research of Computers, 2018, 35 (11): 3206-3212.)

[5] Ko H, Pack S, Leung V. Coverage-guaranteed and energy-efficient participant selection strategy in mobile crowdsensing [J]. IEEE Internet of Things Journal, 2019, 6 (2): 3202-3211.

[6] Yang Jing, Fu Lei, Yang Boran, *et al.* Participant service quality aware data collecting mechanism with high coverage for mobile crowdsensing [J]. IEEE Access, 2020 (8): 10628-10639.

[7] Xiao Mingjun, Gao Guoju, Wu Jie, *et al.* Privacy-preserving user recruitment protocol for mobile crowdsensing [J]. IEEE/ACM Trans on Networking, 2020, 28 (2): 519-532.

[8] Hu Qin, Wang Shengling, Cheng Xiuzhen, *et al.* Cost-efficient mobile crowdsensing with spatial-temporal awareness [J]. IEEE Trans on Mobile Computing, 2021, 20 (3): 928-938.

[9] Song Shiwei, Liu Zhidan, Li Zhenjiang, *et al.* Coverage-oriented task assignment for mobile crowdsensing [J]. IEEE Internet of Things Journal, 2020, 7 (8): 7407-7418.

[10] Zhou Siwang, He Yan, Xiang Shuzhen, *et al.* Region-based compressive networked storage with lazy encodings [J]. IEEE Trans on Parallel and Distributed Systems, 2019, 30 (6): 1390–1402.

[11] Liu Wenbin, Wang Leye, Wang En, *et al.* Reinforcement learning-based cell selection in sparse mobile crowdsensing [J]. Computer Networks, 2019, 161 (9): 102–114.

[12] Liu Wenbin, Yang Yongjian, Wang En, *et al.* User recruitment for enhancing data inference accuracy in sparse mobile crowdsensing [J]. IEEE Internet of Things Journal, 2020, 7 (3): 1802-1814.

[13] Liu Wenbin, Yang Yongjian, Wang En, *et al.* Prediction based user selection in time-sensitive mobile crowdsensing [C]// Proc of the 14th Annual IEEE International Conference on Sensing, Communication, and Networking. Piscataway, NJ: IEEE Press, 2017: 1-9.

[14] Tao Xi, Song Wei. Location-dependent task allocation for mobile crowdsensing with clustering effect [J]. IEEE Internet of Things Journal, 2019, 6 (1): 1029-1045.

[15] Wang Jiangtao, Wang Yasha, Zhang Daqing, *et al.* Fine-grained multitask allocation for participatory sensing with a shared budget [J]. IEEE Internet of Things Journal, 2016, 3 (6): 1395-1405.

[16] Zhou Tongqing, Xiao Bin, Cai Zhiping, *et al.* A utility model for photo selection in mobile crowdsensing [J]. IEEE Trans on Mobile Computing, 2021, 20 (1): 48-62.

[17] Gendy M, Al-Kabbany A, Badran E. Maximizing clearance rate of budget-constrained auctions in participatory mobile crowdsensing [J]. IEEE Access, 2020 (8): 113585-113600.

[18] Xia Xingyou, Zhou Yan, Li Jie, *et al.* Quality-aware sparse data collection in MEC-enhanced mobile crowdsensing systems [J]. IEEE Trans on Computational Social Systems, 2019, 6 (5): 1051-1062.

[19] Nguyen T N, Zeadally S. Mobile crowd-sensing applications: data redundancies, challenges, and solutions [J]. ACM Trans on Internet Technology, 2022, 22 (2) , 1-15.

[20] Wang Liang, Yu Zhiwen, Zhang Daqing, *et al.* Heterogeneous multi-task assignment in mobile crowdsensing using spatiotemporal correlation [J]. IEEE Trans on Mobile Computing, 2019, 18 (1): 84-97.

[21] Zhang Lichen, Ding Yu, WangXiaoming, *et al.* Conflict-aware participant recruitment for mobile crowdsensing [J]. IEEE Trans on Computational Social Systems, 2020, 7 (1): 192-204.

[22] Yang Wenjie, Sun Guodong, Ding Xingjian, *et al.* Budget-feasible user recruitment in mobile crowdsensing with user mobility prediction [C]// Proc of the 37th International Performance Computing and Communications Conference. Piscataway, NJ: IEEE Press, 2018: 1-10.

[23] Yang Yongjian, Liu Wenbin, Wang En, *et al.* A prediction-based user selection framework for heterogeneous mobile crowdsensing [J]. IEEE Trans on Mobile Computing, 2019, 18 (11): 2460-2473.

[24] Wang Jiangtao, Wang Feng, Wang Yasha, *et al.* HyTasker: hybrid task allocation in mobile crowd sensing [J]. IEEE Trans on Mobile Computing, 2020, 19 (3): 598-611.

[25] Zhu Xiaoyu, Luo Yueyi, Liu Anfeng, et al. A deep learning-based mobile crowdsensing scheme by predicting vehicle mobility [J]. IEEE Trans on Intelligent Transportation Systems, 2021, 22(7): 4648-4659.